НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

«До захисту допущено»
Завідувач кафедри
_____ О.І. Ролік
«___»_____2019 р.

Дипломний проект

на здобуття ступеня бакалавра

з напряму підготовки 6.050103 «Програмна інженерія»

на тему: «Інтелектуальна система тайм-менеджменту працівника»

Виконала:
студентка IV курсу, групи ІТ-51
Кісса Олеся Євгеніївна _____

Керівник:
асистент, Дорога-Іванюк О. О.

_____

Рецензент:

_____

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2019 рік

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки – 6.050103 «Програмна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.І. Ролік
«___»_____2019 р.

ЗАВДАННЯ
на дипломний проект студенту
Кісси Олесі Євгеніївни

1.Тема проекту «Інтелектуальна система тайм-менеджменту працівника», керівник проекту Дорога-Іванюк Олена Олександрівна, асистент, затверджені наказом по університету від «___»_____ 2019 р. №_____

2. Термін подання студентом проекту _____

3. Вихідні дані до проекту
_____
_____

4. Зміст пояснювальної записки
_____
_____

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо)
_____
_____
_____

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка

Студент О.Є. Кісса

Керівник проекту О.О. Дорога-Іванюк

АНОТАЦІЯ

Кісса О.Є. Інтелектуальна система тайм-менеджменту працівника. КПІ ім. Ігоря Сікорського, Київ, 2019.

Проект містить 77 с. тексту, 7 рисунків, 5 таблиць, посилання на 18 літературних джерел, 6 додатків та 4 конструкторських документи.

Ключові слова: аналіз тексту, застосунок, машинне навчання, продуктивність, прокрастинація, тайм-менеджмент.

Об’єктом розробки є інтелектуальна система тайм-менеджменту працівника.

Мета розробки — вирішення проблеми управління часом шляхом відстеження виконання завдань та встановлення пріоритетів.

У дипломному проекті було розроблено мобільний застосунок під платформу Android для тайм-менеджменту, шо представляє собою структурований список для організації завдань за пріоритетами та категоріями, які визначаються алгоритмом обробки природної мови.

Отримані результати можуть бути корисними для впровадження на підприємстві з метою підвищення продуктивності або використання готового продукту у повсякденному житті для особистих цілей.

SUMMARY

Kissa O.Y. Intelligent employee time management system. Igor Sikorsky KPI, Kyiv, 2019.

The project contains 77 pages of text, 7 figures, 5 tables, links to 18 literary sources, 6 annexes and 4 design documents.

Keywords: application, machine learning, productivity, procrastination, text analysis, time management.

The object of development is the intelligent time management system for employee’s productivity improvement.

The purpose of the development — solving time management problem by tracking tasks execution and setting priorities.

The graduation project developed Android mobile application for time management which represents a structured list to organize tasks by category and priority that are computed by natural language processing algorithm. The detailed analysis and selection of software architecture for a reliable end product was conducted.

The results obtained can be useful for integration into enterprise sector for productivity improvement or can be used in daily life for personal goals.

Номер рядку	Формат	Позначення	Найменування	Кіл. листів	Номер екзем.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІТ51.330БАК.002 ПЗ	Пояснювальна записка	77		
6	А3	ІТ51.330БАК.003 Д1	Діаграма прецедентів	1		
7	А3	ІТ51.330БАК.004 Д2	Схема реалізації шаблону MVVM	1		
8	А3	ІТ51.330БАК.005 Д3	Блок-схема алгоритму визначення	1		
9			пріоритету			
10	А3	ІТ51.330БАК.006 ТБ1	Матриця відповідності вимогам	1		
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						

					ІТ51.330БАК.001 ТП		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.					Інтелектуальна система тайм-менеджменту працівника Відомість технічного проекту	Літ.	Лист	Листів
Перевір.		.				Т		1
						КПІ ім. Ігоря Сікорського, ФІОТ Група ІТ-51		
Т. контр.								
Затвер.								

Пояснювальна записка
до дипломного проекту
на тему: «Інтелектуальна система тайм-менеджменту працівника»

Київ – 2019 рік

ЗМІСТ

ВСТУП

Швидкість розвитку технологічних нововведень за останні десять років поточного століття може здаватися приголомшливою, проте більшістю людей це вже сприймається як щось само собою зрозуміле, а не технологічне диво або прогрес. Варто зауважити, що інформаційна область стає більш різноманітною: з'являються як нові технології, так і добре забуті, але поліпшені, старі.

Звичайно, інколи те, що відбувається в нашому світі не може не вражати, коли з усіх сторін на людей сипляться такі терміни, як штучний інтелект, нейронні мережі, машинне та глибоке навчання, роботизація завдань, і так далі. Комп'ютерна техніка досягає такої продуктивності та точності, яка дозволяє використовувати її в областях, безпосередньо не пов'язаних зі світом інформації або ж там, де раніше використовувалася тільки людська робоча або інтелектуальна сила, наприклад: генної терапії, авіа- та інших транспортних індустрій, космічного сектору, банківської справи. Підігріті інтересом з боку сучасних медіа-ресурсів і телебачення, молодші покоління працівників намагаються вслідкувати за всіма технологічними тенденціями, особливо це стосується робітників в IT-галузі. Так як компанії, що займаються розробкою програмного забезпечення для комп'ютерів і різних переносних пристроїв, можуть спостерігати очевидну підвищену увагу до даних технологій, кожен на ринку намагається інтегрувати відповідні рішення в готові програми або ж створювати абсолютно нові і наближати їх характеристики під потреби людей.

За результатами різних досліджень і опитувань на ринку інформаційних технологій найбільша кількість програмних продуктів створюється для мобільних телефонів і це здається досить очевидним, бо

потужності процесорів різних моделей можуть змагатися з сучасними процесорами, які розробляються для настільних комп'ютерів і ноутбуків.

Сучасні користувачі зацікавлені в тому, щоб зберігати на відстані руки все, що їм необхідно, в одному пристрої, що є мобільним телефоном. Для виконання персоналізованих потреб і запитів клієнтів якраз і потрібні такі технології, як штучний інтелект і машинне навчання. Використання даних технологій допомагає виробляти відповідний продукт, який допомагає створювати особливе спілкування з кінцевим користувачем, покращує систему пошуку інформації і є найкращим стимулом щоденного використання того чи іншого застосунку.

Метою дипломного проекту є створення інтелектуальної системи з управління часом у вигляді мобільного застосунку, що здатний вирішувати проблему управління часом та дозволяє відстежувати виконання завдань і допомагає користувачеві навчитися виставляти пріоритети для досягнення цілей.

Бакалаврський проект складається з наступних розділів: вступ, постановка задач, аналіз сучасного стану проблеми, аналіз існуючих рішень, реалізація, аналіз готового програмного продукту, висновки, перелік посилань із 18 найменувань, 6 додатків. Графічна частина включає 4 кресленики формату А3. Загальний обсяг 77 сторінок.

1 ПОСТАНОВКА ЗАДАЧ

Метою дипломного проекту є дослідження та розробка інтелектуальної системи тайм-менеджменту, що допомагає із вибором пріоритетів у виконанні завдань, що веде до підвищення продуктивності працівника. Для досягнення мети були поставлені наступні задачі:

- дослідження проблеми тайм-менеджменту;
- проведення огляду існуючих методів аналізу текстів завдань за допомогою використання алгоритмів обробки природної мови;
- розробка інтелектуальної системи аналізу тексту завдань на основі використання алгоритму машинного навчання.

2 АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ

Питання про швидкоплинність часу і про те, чи можна їм управляти, має коріння з давніх часів. Ще давньоримський філософ і поет Луцій Анней Сенека в своєму філософському діалозі "Про короткочасність життя" (лат. De Brevitate Vitae) міркує про те, що природа наділяє людську істоту достатньою кількістю часу на виконання дійсно важливих справ та як індивідууму варто розподіляти відведений йому час, але за різних причин більшість людей це не турбує [1]. У своїй праці Сенека намагається звернутися до людей з фундаментальним питанням про те, як прожити дійсно гідне життя, повне сенсу, й задуматися про умисність відведеного кожній людській особі часу, про те, що насправді не життя коротке, а просто люди схильні займатися марнотратством дорогоцінного часу, в кінці кінців нагадати про те, що життя не може бути вічним. Подібні думки про час також навідували розум грецьких філософів, які трактували його у вигляді двох божеств: Хроноса, який представляв хронологічний календарний час, і Кайроса — вічне, справжнє "зараз" [2]. Кайрос частіше ототожнював появу деякої протоки, через яку потрібно переплисти силою, щоб досягти успіху. У сучасній гонитві за часом люди схильні забувати про Кайрос та його філософське значення і слідують лише за хронологічним часом, фактично відмовляючись від усвідомлення своєї ефективності в поточний момент часу. Є багато речей, вартих уваги, які людські істоти не помічають, тому що сфокусовані на сліпій гонці.

Беручи до уваги всі ці трактати, не можна не замислитися над тим, як сучасний тайм-менеджмент зазвичай не несе в собі філософський сенс, так як усі публікації чи книги з цієї тематики пишуться вже не філософами, а менеджерами або консультантами з різних бізнес-областей. У подальшому це веде до неправильної оцінки та усвідомлення самого часу,

тому багато технік тайм-менеджменту на практиці зазнають крах. Суть проблеми полягає в тому, що гонитва за продуктивністю лише заради самої продуктивності є контр-продуктивним підходом, так як при цьому втрачається істинний сенс подібних засобів управління часом.

Можна нескінченно пробувати нові техніки, методології, прийоми, шукати секрети, щоб встигати робити більше справ за менший період часу і бути більш продуктивним, ефективно виконувати завдання і краще управляти годинами в добі, проте який в цьому сенс, якщо в результаті люди опиняються в ситуації, де вони добровільно здійснюють на себе тиск і в кінцевому підсумку отримують тільки збентеження або розчарування в черговому методі?

Коли людина починає замислюватися про підвищення своєї продуктивності, вона стикається з великою кількістю методик і підходів, і починаючи застосовувати будь-який метод, вона додає все більше і більше завдань в свій список, думаючи, що в цьому полягає сенс продуктивності. Не дивлячись на відчуття зростання рівня продуктивності, в якийсь момент це зростання сповільнюється і велика кількість завдань стає бар'єром, який дуже важко подолати. Подібні наслідки також називаються стратегією саморуйнування, так як через велику кількість постійно створюваних людиною завдань, вона починає намагатися робити більше справ, щоб дотримуватися плану. Через деякий час людина здатна виконувати велику кількість справ, однак це більше не приносить їй задоволення, а навпаки викликає відчуття стурбованості, що вона робить не так. В результаті закрадається інша проблема — проблема перевантаженості і перевтоми, і це не дивно, адже це природна реакція організму, що виникає у відповідь на різноманітні стресові ситуації. Фізіологічно стрес проявляється як захисна реакція організму на будь-яке надзвичайно складне становище чи надмірну

психічну напруженість, в якій знаходиться людина, що намагається працювати нерівномірно понад силу.

Важливо розуміти і завжди пам'ятати про ризики зіткнутися з проблемами зі здоров'ям, адже численні дослідження показують, що високий рівень стресу змушує серце працювати швидше, ніж при звичайних обставинах, що на думку експертів в окремих випадках може призвести навіть до летального наслідку. Звичайно, стрес має різний вплив на людей, проте за словами кардіолога Алана Юна з медичного центру Стенфорда по серцево-судинним захворюванням, хронічний стрес, тобто життя в постійній напрузі, може сильно нашкодити серцю. Подібне нездорове ставлення до роботи в кінцевому підсумку призводить до підвищення артеріального тиску або холестерину.

Крім серцевих захворювань, життя в завантаженому ритмі може призводити до проблем зі сном і порушень когнітивних функцій мозку (таких, як сприйняття та обробка інформації про навколишній світ, запам’ятовування і зберігання цих даних), що було доведено в різних дослідженнях: наприклад, дослідники Університету Гронінгена стверджують, що при постійній завантаженості частина мозку, що називається гіпокамп та відповідає за навчання, емоції й консолідацію пам'яті, а саме перехід короткочасної пам'яті в довготривалу, починає зменшуватися [3].

У вищезазначеному дослідженні результати також показали, що зміна розміру гіпокампу сильно впливає на ефективність в процесі навчання і настрій людини. Один з дослідників, нейробіолог Пітер Меерло стверджує, що у пацієнтів, яким був поставлений діагноз депресії, також була зменшена дана область мозку та порушена система виділення певного нейротрансмітера — серотоніну, який часто називають "гормоном щастя".

Це призводить до висновку, що проблеми зі сном можуть бути не тільки симптомами такого захворювання як депресія, але і бути її причиною.

Крім зменшення гіпокампу, стрес також може пошкодити нейрони головного мозку, що знаходяться в префронтальній корі — це частина мозку, що відповідає за вирішення проблем, процес пристосування до вирішення складних завдань, управління емоційним комплексом людини і регуляцію обміну речовин.

Дослідження психіатра Раджіта Сінха та її колег з Єльского Університету показують, що переживання і стресові ситуації тягнуть за собою розвиток психічних розладів і можуть служити сигналами появи хронічних хвороб, таких як гіпертонія і діабет [4]. Так як раніше зміни в корі головного мозку вже пов'язували зі стресом, дослідники підтвердили, що люди з великою кількістю подій в житті більш схильні до цих змін, ніж їх менш завантажені колеги.

Також одним з найсерйозніших наслідків стресу може бути тяга до вживання алкогольних речовин. Дослідження, проведене професором Маріанною Віртанен в Фінському інституті гігієни праці, охоплює дані більш ніж 330 000 людей і результати показують, що спостерігається кореляція між завантаженим ритмом життя і вживанням алкоголю [5]. Більш тривалий робочий тиждень (48 годин роботи) тісно пов'язаний з більш інтенсивним вживанням алкоголю незалежно від статі працівника, його географічного положення або приналежності до соціально-економічної групи.

Одним з факторів, який виникає внаслідок розвитку стресового стану у співробітника і здатний порушити робочий процес — це втрата уваги. У даній ситуації співробітник зазнає зниження концентрації, що позначається на його працездатності. Концентрація при роботі дуже

важлива: при виконанні будь-якого завдання для досягнення мети необхідно ефективно зосереджувати увагу саме на виконуваній задачі.

При добре розвиненому навику концентрації людина здатна блокувати різні відволікаючі фактори, що суттєво полегшує робочий процес. Також в подібному стані співробітник менш схильний робити помилки, витрачає меншу кількість часу на виконання поставленого завдання і краще обробляє і запам'ятовує інформацію, з якою він працює. В іншому випадку, стрес впливає на робочий процес: у співробітника знижується продуктивність, він більш схильний робити помилки в різних аналітичних ситуаціях або завданнях, що вимагають повної віддачі і концентрації. У подібних ситуаціях людина не може не відволікатися на речі, ніяк не пов'язані з роботою.

У 2012 році, Мая Салавіц, американський журналіст і дослідник в області лікування різних видів залежностей, провела дослідження, що стосується поведінки людини в стані стресу [6]. Дослідження показало, що коли людина приймає важливі рішення в стані стресу, вона більш схильна до відволікання і концентрації на непотрібних в момент виконання складного завдання речах. Даний процес відбувається в результаті того, що мозок змінює підхід до прийняття рішень і на шальках терезів зазвичай переважує короткочасне задоволення, а негативні наслідки від подібної поведінки ігноруються або ж відкладаються на другий план на невизначений термін. У стані стресу людина схильна здійснювати будь-які дії, які можуть моментально принести задоволення, але які потім приносять безліч проблем в довгостроковій перспективі.

Вищезазначена поведінка, при якій людина схильна уходити з головою в короткострокові задоволення, не беручи до уваги той факт, що це може нести за собою негативні наслідки, також призводить до ефекту накопичення прокрастинації. Даний термін не раз зустрічається в

різноманітних статтях з психології тайм-менеджменту і це не дивно, адже прокрастинація безпосередньо пов'язана зі стресом.

Сам термін "прокрастинація" з'являється ще в самих ранніх текстах писемностей, які налічують не менше 3000 років [7]. Деякі з перших записів, наприклад керівництва по землеробству, описували це як суттєву проблему. Згадування проблем прокрастинації також з'являлися в ранніх римських і грецьких військових документах, і в старовинних церковних і релігійних текстах. Даний термін має безліч трактувань, що несуть як позитивний, так і негативний посил, однак найбільш доступним і популярним визначенням прокрастинації є поняття, на яке посилається в своїй науковій роботі Пірс Стіл, дослідник в галузі науки мотивації і прокрастинації: "Прокрастинація це особливий вид відкладання справ на потім; прокрастинація нераціональна" [8]. У різних словниках дане поняття трактується як "навмисне відкладання виконання справи, яке повинно бути зроблено".

Прокрастинація не завжди може мати негативний посил, все залежить від ситуації. Наприклад, якщо свідомо відкладати виконання будь-якої задачі, тому що вона не є пріоритетною та вважати, що це допоможе кінцевому результату, то таке відкладання справ не зовсім можна охарактеризувати як прокрастинацію. Однак іноді поняття прокрастинації використовують не за призначенням. Зазначений випадок з відкладанням справи на потім в ситуації, де це може допомогти кінцевому результату, краще охарактеризувати як “передбачливість” або ж “розсудливість”, але ніяк не “прокрастинація”.

Однак же, якщо людина планує виконання завдання в останній момент або відчуває, що це потрібно було зробити раніше, а потім все одно відкладає справу, то це вже явно можна назвати прокрастинацією. В цілому, це процес відтягування справи навіть тоді, коли індивідуум усвідомлює, що

дана поведінка призведе до негативних наслідків, тобто, він усвідомлено уникає роботи і навмисне шукає будь-яке відволікання, аби не виконувати поставлене завдання.

У більшості випадків, прокрастинація — це відображення проблеми підвищеного відволікання, низької мотивації, нездатності до саморегуляції і невміння адекватно передбачити самопочуття при негативних наслідках. Усі прокрастинатори (люди, схильні відкладати справи на потім) зазвичай розподіляються на три категорії: шукачі гострих відчуттів (англ. Thrill seekers), втікачі або авойдери (англ. Avoiders) і перфекціоністи (англ. Indecisives) [9].

Шукачі гострих відчуттів люблять приступати до виконання завдання, коли крайні терміни здачі роботи на підході. Подібні люди відчувають, що вони краще справляються з завданнями, добровільно піддаючи себе стресу, але самі не усвідомлюють, на яку небезпеку наражається при цьому організм.

Втікачі — дана група людей ухиляється від прийняття рішень і будь-яких дій, так як вони бояться кінцевого результату, яким би він не був, гарним або поганим. Багато речей, яких люди намагаються уникати, зазвичай відносяться не до самої проблеми, а до емоцій і почуттів, які відчуває людина в процесі виконання завдання або ж при його завершенні, коли потрібно зіткнутися віч-на-віч з результатом виконаної роботи. Також це є психологічним трюком, коли людині легше не приступати до виконання завдання, ніж потім зіткнутися з потенційною невдачею при її виконанні.

Перфекціоністи — прокрастинатори, які зазвичай відкладають справи на потім з метою зняти з себе відповідальність або ставлять нереальні вимоги до кінцевого результату, що призводить до витрати величезної кількості робочого часу на планування роботи, коли можна приступити до її виконання. Нераціональне ставлення до погляду на кінцевий результат

призводить до певної форми насильства над собою, бо досконалості, як правило, не існує, тому гонитва за ідеальним результатом, а не фокусування на самому процесі виконання завдання, є тратою часу.

Розуміння різних форм прокрастинації допомагає здійснити більш результативний підхід до вирішення тієї чи іншої проблеми і уникнути повторення помилок при плануванні, скоєних раніше.

Беручи до уваги усі захворювання і розлади, які можуть спричинити перенапруження та стрес через неправильний розподіл часу або перевантаження через нераціональне сприйняття реальності, таке як "виконувати більше завдань — означає бути продуктивним" не можна не зробити висновок, що подібний підхід до управління часом може тільки повернути людину до вихідної точки і питання чому вона почала цим займатися. Це призводить до втрати мотивації і рушійної сили, і внаслідок людина розчаровується у застосованій методиці.

Чому ж на практиці велика кількість технік по тайм-менеджменту терплять крах? Існує думка, що з більшою свободою вибору в розподілі особистого часу приходить більша відповідальність, яка накладає певний стрес на співробітника, так як не всі можуть зробити це правильно навіть не з першої спроби. На підприємствах подібні проблеми рівномірного розподілу завдань за часом виконання керівництво раніше брало на себе.

Наприклад, в XX столітті багато співробітників, які працювали на фабриках, не переймалися процесом розподілом часу, тому що навіть час роботи за конвеєром вже був розподілений для них. Пізніше на різних заводах почав з'являтися так званий "технологічний тайм-менеджмент", де використовувалися інструкційні карти для робітників, але із розвитком техніки прихильники наукової організації праці залишили дані картки у минулому. Кожен фахівець в різних областях почав розглядатися як активне джерело вдосконалення прийомів роботи або ж підвищення ефективності

підприємства в цілому. Згадані раніше інструкційні картки робочих складалися для окремих операцій із загального робочого плану та розподілялися за рівнем складності.

Якщо взяти за приклад області механіки і машинобудування, такі картки повинні були містити детальні вказівки, що роз'яснюють процес виконання складної складальної операції або ж забезпечувати робочих докладними картами інструкцій контролю, що встановлюють єдині вимоги, яким повинен задовольняти об'єкт. Усередині карти докладно вказувався метод перевірки того чи іншого елемента об'єкта в тій послідовності, якої слід дотримуватися контролерові або складальникові в своїй роботі. Інструкційна карта поділялася на різні категорії, що містять технічні установки і вид робіт, який повинен виконати працівник. Обов'язковим атрибутом також був календарний план, за яким відстежувалося виконання робіт і графік завантаження підприємства в цілому. При подібній техніці планування часу досягалося якісне виконання роботи, без зривів крайніх термінів здачі результатів відповідному керівництву, а також забезпечувалася планова перевірка і написання звітностей про виконану роботу.

Приведений підхід до підвищення ефективності праці не тільки не втратив свою актуальність в XXI столітті, але і став незамінним засобом управління часом, коли контроль виконання завдань застосовується не тільки для масового виробництва, де необхідно точно контролювати трудові процеси, але також використовується індивідуальними співробітниками. Більшість співробітників в різних фірмах і офісах сьогодні мають достатню свободу в реалізації самих себе і контролі робочим часом, однак це призводить до великого тиску. Виникає багато питань, про те, як краще розподілити поточні завдання, як навчитися розставляти пріоритети без шкоди для ефективності робочого процесу та власного самопочуття.

Багатьом співробітникам також доводиться поєднувати під своїм керівництвом не один проект, що часто позначається на інших сферах життя, таких як соціальне життя і сім'я.

Причин, за якими варто звернути особливу увагу на управління часом, досить багато, проте необхідно виділити найосновніші:

— оволодіння новими навичками для задоволення особистих потреб або ж підвищення продуктивності в робочій області;

— порушені терміни здачі проектів через невміння виставляти пріоритети;

— усвідомлення відчуття втоми або стресу, коли людина нібито весь день була зайнята, але в кінцевому підсумку нічого не зробила, тобто був отриманий нульовий результат — так званий синдром удаваної зайнятості.

Варто детальніше розглянути синдром удаваної зайнятості. При цьому синдромі людина схильна не випускати з рук гаджет і це стає нормою її життя. Негативний вплив створює велика кількість зовнішніх подразників (в даному контексті це можуть бути різні застосунки, які ніяк не допомагають у просуванні виконання завдання, тільки у зворотну сторону), що ведуть до зниження концентрації і підвищення відволікання від роботи, яку потрібно виконати в термін.

Робочий день у такого співробітника є ненормованим, часто закінчується на кілька годин пізніше, ніж реально потрібно на виконання деякого завдання. Чим важче завдання, тим нудніше воно зазвичай здається, з'являється неуважність через множинні відволікання і збільшується кількість спроб уникнути виконання завдання. Якість виконання роботи в цілому знижується, розвивається хронічна втома і у деяких випадках навіть депресія. Уміння фокусуватися на будь-якій задачі — необхідна навичка поліпшення продуктивності.

Деякі люди думають, що бути зайнятими чимось, означає бути продуктивними, проте в цьому і полягає проблема удаваної зайнятості. Тому так важливо навчитися балансувати між поставленими завданнями і стежити за витратою і відновленням особистої енергії. Цей підхід допомагає стимулювати високу продуктивність.

В основі контролю часу лежить грамотне планування за допомогою різних способів, таких як ті ж застосунки в гаджетах, з якими не розлучається людина в сучасному світі, що здатні спрямувати її потенціал в потрібне русло, наприклад підлаштувати час під управлінські цикли на підприємстві.

Висновки: в умовах швидкого розвитку суспільства і щоденного отримання великого обсягу інформації людині потрібно все більше часу, щоб використовувати надані ресурси в повному обсязі. Прискорення темпів життя передбачає будування найбільш раціональної та ефективної діяльності за допомогою тайм-менеджменту. Головне завдання, яке ставить перед собою тайм-менеджмент — це організація планової роботи, розподіл часового ресурсу, вміння виставляти пріоритети для виконання більшої кількості роботи за менший період часу та відстежувати отримані результати. Концепція управління часом це не просто різні підходи і методи, — це філософія ставлення до життя і усвідомлення цінності часу. Грамотне планування часу із правильно виставленими пріоритетами та обраними першочерговими задачами приносить свої плоди у вигляді своєчасної реалізації поставленої цілі та здачі роботи, при цьому якісно виконаної. Управління часом має бути невід’ємною звичкою сучасної людини задля досягнення успішності та запобігання чисельних проблем зі здоров’ям та соціумом.

3 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

Люди схильні структурувати інформацію за допомогою створення списків. Дана методика підходить для спрощення сприйняття інформації, складних концепцій, для звільнення свідомості від зайвих думок, тощо. Створення списків допомагає краще зрозуміти навколишній світ через письмову форму.

Творцем і основоположником ведення справ у вигляді списку вважають Бенджаміна Франкліна, який був вченим, винахідником, журналістом і політичним діячем. Франклін відомий тим, що свого часу вирішив створити тринадцяти-тижневий план по саморозвитку у вигляді списку, де планував практикувати різні чесноти, такі як охайність, стриманість, ощадливість і т.д., і в який він щодня вносив прогрес і зміни, і відображав це все на графіку [10]. Бенджамін Франклін створив для себе суворий графік, де окрім роботи по пунктам були розписані навіть такі активності, як сон та прийоми їжі. Кожному пункту відводився відповідний проміжок часу.

Якщо раніше люди користувалися рукописними або друкованими варіантами структурування інформації та завдань, наприклад інструкційними картками працівників або паперовими списками, то в еру розвитку інформаційних технологій більшість списків існує і ведеться саме в електронній формі. Це можуть бути як вебсайти, так і мобільні застосунки, які є більш поширеним методом ведення справ, так як доступ до мобільного телефону є практично завжди. Будь то список покупок або список справ, використання списків в мобільних застосунках є дуже зручним методом поліпшення продуктивності.

Існує багато технологій та методик з тайм-менеджменту, проте не всі вони дійсно можуть застосовуватися в мобільних застосунках. На сьогоднішній день найбільш реальними в плані реалізації саме на мобільних платформах є технології “Впоратися зі справами” (англ. Getting Things Done,

GTD), метод Помодоро та побудова так званого “ланцюга” справ або правило “Don’t break the chain”.

3.1 Getting Things Done

Методологія “впоратися зі справами", автором якої є Девід Алан і яка є однією з найпопулярніших технік протягом багатьох років, є однією з кращих систем по поліпшенню і підтримці продуктивності особистості. Спочатку вона може здаватися складною через незвичність, проте кінцевий результат виправдовує всі очікування. Головною метою є витрачати менше часу на неприємні або ж важко-виконувані завдання, а більше часу відводити на те, що людині дійсно до душі. Основним принципом тайм-менеджменту в цій методології є розподіл завдань по пріоритетам, створення такого графіка, при якому можна виконати завдання в строк, причому кількість завдань вибирається раціонально. Особливо зручним є те, що дана методологія дозволяє охопити поглядом весь список справ, так як він є структурованим і потім вибирати те завдання, над яким за думкою людини варто працювати далі. Головне — завжди записувати те, що приходить в голову, щоб звільнити мислення від зайвих процесів і потім фокусуватися виключно на виконанні завдання. Даний підхід дозволяє очистити розум від будь-яких відволікань, які перешкоджають ефективній роботі.

Люди, що використовують дану методологію, посилаються на її складність у зв'язку з тим, що вони не обмежені підходами, які вони повинні використовувати при її застосуванні. Є певні установки щодо самого занотовування справ, але ніхто не обмежений у виборі

додаткових засобів для здійснення задуманого. Цими установками є:

— записувати все, що приходить на розум, тобто не відкладати це на потім, тому що інформація швидко зникає з голови;

— уточнювати дії і уникати абстрактних формулювань, наприклад якщо необхідно написати місячний звіт про продаж N одиниць товару в конкретному сегменті ринку, то не варто вносити в список скорочену фразу "написати звіт";

— організовувати завдання по категоріях і за пріоритетом — це необхідно для виконання дійсно важливих завдань. Корисним також буде написання крайніх термінів здачі того чи іншого завдання;

— періодично переглядати список для відстеження прогресу або ж уточнення крайніх термінів завершення роботи — це допомагає виконувати роботу в термін й при потребі прибирати зі списку речі, які не є пріоритетними на даний момент;

— після перегляду списку відразу ж приступати до виконання завдання, на даному етапі список структурований, кожне завдання має свою категорію і пріоритет, при потребі розбита на підзадачі, і залишається тільки вибрати те, над чим потрібно працювати.

Не існує правил, які необхідно застосовувати в плані самоорганізації, використовуючи дану методологію, тобто завантажувати спеціальний застосунок або купувати блокнот для записів. В кінцевому рахунку, це може бути навіть звичайний аркуш паперу, на якому олівцем або ручкою пишеться список справ, а потім вони закреслюються по мірі завершення.

3.2 Метод Помодоро

Даний метод був розроблений Франческо Чірілло ще в кінці 1980-х для боротьби з неуважністю під час роботи [11]. Метод називається саме “помодоро”, тому що Чірілло використовував таймер у вигляді помідора для відстеження робочого процесу, коли він ще сам був студентом університету. Основний принцип полягає в тому, щоб виконувати тривалу роботу інтервалами протягом 25 хвилин, між якими робляться короткі перерви на 3-5 хвилин, щоб подихати повітрям або відволіктися на щось радіально протилежне, що дасть мозку кращий стимул для виконання роботи в наступному інтервалі. Метод Помодоро є ітеративним методом, тобто його можна виконувати в кілька підходів, але після кожного четвертого 25-хвилинного інтервалу варто робити перерву тривалістю від 15 до 30 хвилин, що зміцнює концентрацію пізніше і підвищує креативність. Метод Помодоро, ймовірно, є одним з найпростіших способів підвищення продуктивності. Все, що потрібно для його реалізації, це таймер. Не важливо, буде це настільний таймер, або електронний таймер в мобільному застосунку, важливо дотримуватися інтервалів. Варто відзначити, що якщо з якої-небудь причини людина відволікається від виконання завдання, то вона зобов'язана або тут же зупинити таймер, або ж відкласти термінову справу до того, як час на таймері не закінчиться.

3.3 Правило “Don’t break the chain”

Наостанок варто розглянути правило, висунуте Джеррі Сайнфелдом, американським гумористом та актором. Ідея полягає

в тому, щоб повісити на стіну величезний календар, де на одній сторінці будуть розташовані всі дні року, і запастися червоним маркером. При виконанні якоїсь певної задачі або ж цілого списку справ потрібно закреслювати день великим знаком X. Після кількох днів дотримання даного правила на календарі можна буде побачити щось, схоже на ланцюжок.

Дотримуючись цього порядку, людина буде намагатися не порушувати цей ланцюжок, так як він візуально приваблює і створює відчуття підвищення продуктивності. Даний метод працює, тому що він допомагає виробити постійність у виконанні завдань. Це може бути поліпшення навичок писемності, вивчення нової мови або ж звичайні домашні клопоти, якщо робити це постійно, у людини виробляється звичка та певна активність перестає здаватись тягарем.

Альтернативою великому паперовому календарю з маркером може послужити функція в застосунку з тайм-менеджменту, що дозволяє помічати завдання як виконані і розміщувати їх в спеціальний відділ, наприклад архів, щоб людина могла візуально спостерігати результати виконаної роботи. Даний підхід підтримує інтерес людини у збереженні певного рівня продуктивності.

Був проведений детальний пошук і аналіз існуючих рішень у вигляді мобільних застосунків, які можуть допомогти привести в дію наведені технології управління часом, серед яких головними критеріями оцінки продукту були:

— зручність і простота інтерфейсу;
— вартість;
— відповідність вимогам користувача.

Найбільш популярними застосунками на ринку виявилися Todoist, Checklist і Tasks. У даних продуктах присутні загальні характеристики, про які користувачі залишали позитивні відгуки на відповідних платформах, а саме додавання нових завдань, їх редагування та видалення, налаштування системи оповіщень, можливість додавати замітки до завдань. Однак варто навести детальну описову характеристику про кожний програмний продукт, щоб виявити його сильні та слабкі сторони.

3.4 Todoist

Цей застосунок є вибором редакції компанії Google в 2019 році, за словами вебсайту The Verge: "Todoist — кращий застосунок для створення списку задач", також володіє відмінним функціоналом, необхідним для тайм-менеджменту, а розробники позиціонують його як інструмент, який може зробити практично все, що потрібно користувачеві.

В даному застосунку можливо створювати різні види списків: списки справ, покупок, так званий бакетлист (англ. Bucket list), до якого вносяться речі або справи, які людина бажає зробити до кінця життя; організовувати управління проектом, за допомогою створення окремого проекту і призначення завдань за проектом конкретної людини — виконавця; виставляти пріоритети завданням і налаштовувати оповіщення з можливістю їх періодичного повторення. Синхронізація на різних пристроях дозволяє відстежувати прогрес на великому екрані монітора і надає непоганий огляд всіх існуючих завдань. Графічний інтерфейс не є перевантаженим, виглядає лаконічно і просторо. Наприклад, в застосунку домінує червоний колір, який наповнює дизайн

продукту важливістю. Даний колір також використовується, якщо користувачеві потрібно звернути належну увагу на який-небудь елемент, в даному додатку такими елементами виступають лейбли завдань, графіки з прогресом користувача, кнопки, дати виконання, виконавці. З іншого боку, це не зовсім зручно, так як зазвичай необхідно більше кольорів для відмінності тих же елементів контролю та інформаційних елементів.

Переходячи до недоліків, варто для початку відзначити вже згадані кольорові акценти: розподіл різних акцентів для елементів є гарною практикою, так як у користувача немає відчуття, що всі елементи знаходяться в одній купі, якщо вони мають однаковий відтінок. Для кращого сприйняття структури інтерфейсу можна скористатися колірним кругом для вибору комплементарних кольорів, які знаходяться один навпроти одного і підкреслюють один одного, не створюючи при цьому перевантаження відтінків, так як при змішуванні вони дають нейтральний сірий колір.

Одним із суттєвих недоліків також є необхідна реєстрація для використання застосунку. Хоч застосунок і надає можливість авторизації за допомогою Google або Facebook аккаунта, це додає надлишкові кроки і додатковий обсяг інформації, який потрібно запам'ятати людині, перш ніж з'явиться можливість протестувати існуючий функціонал. Взагалі, дослідження в області розробки мобільних застосунків показують, що причина наявності реєстрації, а точніше її довгий або складний процес, є на другому місці в списку причин, за якими користувач може вирішити не використовувати продукт [12].

При першому запуску застосунку Todoist після проходження реєстрації та авторизації з'являється екран зі складеним базовим

списком, який є статичним і містить вказівки щодо функціоналу та деякі графічні елементи. Це не зовсім зручний підхід, так як сучасні користувачі звикли до анімаційної навігації при першому запуску, яка допомагає їм ознайомитися з основним функціоналом. Інтерфейс виглядає лаконічно, однак при відкритті одного з меню (яких в даному додатку два) з'являється величезна кількість опцій і в цілому зручність використання є досить негативним через згаданий статичний показ.

Також не дуже зрозумілим є розділ Inbox, куди можна записувати завдання, для яких складно визначити категорію, так як логічно можна припустити, що дана опція є поштовою скринькою через схожість назви з іншими сервісами, такими як наприклад Gmail від компанії Google або ж Microsoft Outlook від Microsoft.

Багато функцій застосунку виявляються доступними тільки після придбання преміум-аккаунта, а саме система оповіщень, можливість додавати замітки до задач і лейбли, можливість відстежувати статистику виконаних завдань і вивчати графік продуктивності користувача, та збереження завдань на акаунті. Тож не дивно, що в цьому застосунку досить негативних відгуків, так як багато користувачів звикли до існуючого базового функціоналу в інших додатках. Якщо взяти за приклад можливість ставити нагадування про поточні справи або події, то це невід'ємний інструмент відстеження, і надавати його тільки за окрему плату не є коректним, особливо після того, як на офіційному сайті застосунку в розділі "Як це працює" згадана можливість "ніколи не випускайте з уваги важливе завдання". Це може викликати почуття того, що людину ошукали і привести до невиправданих очікувань з боку користувача, через що вірогідність того, що він повернеться

до користування даним додатком, дуже низька. Також за відгуками застосунок може перенести створений проект в абсолютно іншу папку або інший розділ без відома користувача. Без існування докладної інструкції з використання, користувачам важко розібратися в функціоналі і деякі дії, які вони можуть здійснювати інтуїтивно, такі як змахування елементів для їх видалення або ж довге натискання на елемент для появи контекстного меню з різноманітними опціями, просто відсутні в даному застосунку.

3.5 Checklist

Цей застосунок є засобом для управління часом і позиціонується як застосунок, безкоштовний на 100% з усім необхідним функціоналом і без преміум версій. Основними функціями є створення списків, можливість зберігати створені списки в хмарному сервісі, синхронізація між пристроями, призначення виконавців для кожного конкретного завдання, можливість прикріплювати файли до завдань і отримувати сповіщення.

Графічний інтерфейс виглядає приємно на перший погляд. В даному застосунку в якості основного кольору використовується блакитний, який є одним з найбільш популярних кольорів при розробці дизайну для продукту і також вважається одним з основних додаткових кольорів, тобто це один з відтінків, який при змішуванні дає білий колір. Даний відтінок несе в собі сенс важливості і впевненості, але не надто перевантажує увагу. Також він символізує безпеку і довіру, що дуже важливо, так як користувач повинен відчувати беззаперечну довіру до продукту,

який збирається використовувати для планування такого цінного ресурсу, як власний час.

Однією з переваг застосунку є можливість використання шаблонних списків, запропонованих системою. Це певний набір раніше створених списків, розподілених на деякі категорії, наприклад "Baby Checklist" для догляду за дитиною.

Усередині застосунка Checklist можна виконати пошуковий запит в системі Google або ж перейти до швидкої купівлі предметів на сервісі Amazon, проте щоб знайти цю функцію, необхідно заходити в контекстне меню кожного з елементів. Це є істотним недоліком, так як дана опція повинна бути винесена в загальне бокове меню для швидкого доступу. Також існує два режими роботи зі списком: стандартний і просунутий, при виборі якого відкриваються додаткові опції в контекстному меню елемента списку.

З подальшим зануренням в дослідження функціоналу програми були виявлені такі недоліки, які суттєво позначаються на зручності використання:

— для отримання доступу до функціоналу необхідна реєстрація, як і в випадку із застосунком Todoist;

— запропонована функція авторизації за допомогою облікового запису Facebook присутня на екрані з авторизацією користувача в системі, але при спробі використати аккаунт з’являється модальне вікно з повідомленням про те, що ця функція не підтримується;

— після першого запуску застосунку процес ознайомлення з функціоналом недостатньо докладний, безліч функцій і їх призначення доводиться вгадувати емпіричним підходом;

— список із завданнями рухається по осі X в обидві сторони і не займає весь робочий простір екрану, що призводить до нелогічного сприймання розташування елементів на екрані;

— при відмітці завдання як виконане, елемент переноситься вниз списку, де продовжує займати місце на екрані, і для видалення конкретного елемента необхідно користуватися додатковою функцією з контекстного меню або заздалегідь налаштовувати сортування та фільтрацію завдань у відповідній опції меню, як показано у керівництві користувача на офіційному сайті застосунку. Варто зазначити, що у самому додатку ця функція відсутня за замовчуванням і треба виконати не зовсім логічні дії, щоб функцію можна було побачити та скористатись нею;

— застосунок позиціонує себе як повністю безкоштовний, проте при виборі опції "Trigger a template" (побудова списку за шаблоном) пропонується придбання преміум-версії Checklist Pro для можливості створення проектів та їх управління за допомогою призначення виконавчих ролей.

В цілому, після випробування функціонала, застосунок викликає спірні відчуття. Досить велика кількість користувачів також негативно відгукувалася про неясність інтерфейсу, раптові видалення цілих списків без можливості відновлення, тому що немає відповідного модального діалогу з підтвердженням видалення, повільна синхронізація зі списками, які попередньо призначені для сумісного використання між людьми, і з хмарним сервісом.

3.6 Tasks

Цей застосунок має необхідний базовий функціонал і відрізняється від двох попередніх своєю простотою та зручністю використання. Варто відзначити, що Tasks дозволяє приступити до роботи відразу після встановлення застосунку, не вимагаючи користувача проходити реєстрацію або авторизацію в системі, що є дуже зручним підходом до швидкого записування завдань. Подібна практика також дозволяє використовувати на максимум технологію продуктивності GTD, описану в розділі 3.1. На прикладі даного застосунку принцип полягає в тому, що як тільки думка приходить в розум людини, вона у той же момент повинна звільнити голову за допомогою перенесення цієї думки на зовнішній носій, наприклад записавши це в електронний список.

Що стосується графічного інтерфейсу, в цьому застосунку він мінімалістичний, до того ж вибір кольорової гами для кожного нового списку цілком належить користувачеві. Подібний підхід сміливо можна назвати відмінним ходом в можливості надання максимально комфортного інтерфейсу для кожного окремого користувача, так як він може налаштувати вигляд застосунку, грунтуючись на особистих уподобаннях і сприйняттях кольору, а не використовувати заздалегідь створену кольорову гамму.

З переваг також варто відзначити можливість виставляти пріоритети завданням, меню не є перевантаженим, його пункти досить зрозумілі, щоб збагнути яка опція за що відповідає. Також можна переміщати елементи по списку, проводячи таким чином реорганізацію завдань, і між списками, що є зручним функціоналом.

Незважаючи на велику кількість позитивних характеристик, в цьому застосунку існують свої недоліки, а саме:

— немає автоматичного розподілу завдань за категоріями;

— інтуїтивні жести присутні, однак змахування в обидві сторони призводить до видалення елемента, що несе в собі певну невизначеність, так як зазвичай за найкращими практиками радять, що один жест відповідає одній конкретній дії, наприклад: змахування елемента вліво призводить до його видалення, а змахування вправо — до позначки завдання як виконане).

Як підсумок варто зазначити, що у застосунка Tasks набагато більше позитивних відгуків у порівнянні із попередниками. Найбільш за все люди цінують простоту використання, інтуїтивність інтерфейсу, можливість саморуч налаштовувати кольорову гамму, можливість створювати і зберігати різні списки в одному застосунку, замість того, щоб завантажувати різні застосунки для конкретних цілей.

Висновки: мобільні застосунки для створення списків слугують відмінним способом навчитися управляти своїм часом за допомогою структурування інформації у вигляді створення списків, виставлення пріоритетів та налаштування системи оповіщень про важливі події. Хоч вони і мають систему оповіщень, ні в одному з них не можна виставити таймер на виконання роботи, що є великим недоліком усіх розглянутих застосунків. Розглянуті застосунки є незамінними електронними помічниками, які знаходяться на відстані витягнутої руки і допомагають звільнити розум людини від інформації, структуруючи її в зручному для сприйняття вигляді, дозволивши їй таким чином фокусуватися на самому процесі виконання справ, а не на думці про те, в якому обсязі вони зберігаються в голові або намагатися уникнути їх виконання.

4 РЕАЛІЗАЦІЯ

4.1 Обґрунтування вибору платформи розробки та мови програмування

Для реалізації інтелектуальної системи тайм-менеджменту була обрана мобільна платформа. Даний вибір ґрунтується на статистиці, наданій веб-ресурсом Statista, який є провідним постачальником ринкових і споживчих даних в більш ніж 50 країнах світу. За даними ресурсу, індустрія мобільних застосунків

квітне, нараховуючи близько 2.7 мільярда користувачів смартфонів по всій земній кулі [13]. Для порівняння: в 2018 році кількість продажів смартфонів в світовому масштабі дорівнювала 1.56 мільярда, тобто при чисельності населення світу більше 7.691 мільярда чоловік, 20% від цього числа набуло новий смартфон в минулому році. У 2018 році 52.2% всього трафіку веб-сайтів в світі було згенеровано за допомогою мобільних телефонів.

Світовий постачальник аналітики маркетингу в індустрії ігор і мобільного ринку Newzoo регулярно публікує статистику по країнах і регіонах, в якій представлені звіт і прогнози використання мобільних пристроїв. Останній опублікований звіт у вересні 2018 року показує, що поширення смартфонів в Україні досягає відмітки в 48.3% від загального населення країни, що є досить високим показником того, що попит на мобільні пристрої є і він чималий [14].

Мобільні застосунки грають життєво важливу роль в житті людей, як в робочому, так і в особистому. Інтернет-видавництво TechCrunch надає цікаву статистику: людина в середньому використовує 9 мобільних застосунків в день (причому на використання саме мобільних застосунків йде близько 90% часу від загальної кількості часу користування мобільним пристроєм), а в місяць ця цифра досягає 30 [15]. Мобільні застосунки є домінуючою формою взаємодії між бізнесом і кінцевими користувачами. Сучасні користувачі постійно перебувають у русі, так як ритм життя в цифровому столітті неймовірно швидкий, тому так важливо зберігати всю інформацію під рукою. У 2018 році по всьому світу було завантажено 194 мільярди застосунків, що є вагомою характеристикою для створення мобільного застосунку.

На даний момент на ринку мобільної індустрії є дві провідні платформи, це Android від компанії Google та iOS від компанії Apple. Тільки в 2016 році кінцевим користувачам по всьому світу було продано майже 1.5 мільярда смартфонів з операційними системами Android або iOS. Android, на який припадає 81.7% усіх продажів смартфонів, є лідером на ринку. За даними вебсайту Digital Trends можна спостерігати наступну статистику загальної кількості застосунків, які поставляються на кожну з платформ, Google Play Store й Apple App Store відповідно [16]:

— Android apps: 3.5 мільйона застосунків;

— iOS apps: 2.2 мільйона застосунків.

Спираючись на перераховані вище дані й статистики з різних джерел, платформа Android була обрана для створення мобільного застосунку для тайм-менеджменту.

На сьогоднішній день існує дві офіційні мови, які підтримуються платформою Android для написання різних сервісів і застосунків, а саме мови програмування Java і Kotlin. До 2017 року Java була єдиною підтримуваною мовою програмування для розробки, проте потім з'явився Kotlin, синтаксис якого грунтується на мові Java і трохи полегшує деякі аспекти розробки. Однак, так як ця мова була впроваджена в розробку відносно недавно, вона все ще не така популярна і набагато важче знайти відповіді на питання, якщо виникають труднощі з будь-яким модулем мови. Для порівняння можна поглянути на питання на ресурсі Stack Overflow з тегами Java (1.55 мільйонів) і Kotlin (23 тисячі). Програми написані мовою Kotlin займають частку ринку порядку 4.81% на Google Play Store, що є досить малим числом. Беручи до уваги всі перераховані чинники, вибір був зроблений на користь мови програмування Java.

4.2 Проектування архітектури додатку

Для побудови ефективної і надійної системи, яка буде зручна в супроводі, підтримуванні і тестуванні, необхідно розуміти, що така система повинна володіти внутрішніми характеристиками, грамотно спроектованими для підтримки у подальшій розробці, і зовнішніми характеристиками, які орієнтовані на кінцевих користувачів і які повинні представляти максимальну простоту використання продукту.

4.2.1 Принципи SOLID

Так як обрана мова програмування Java є об'єктно-орієнтованою, необхідно розглянути такий підхід до проектування надійної системи, який буде забезпечувати її підтримку в надійному вигляді, шляхом використання п'яти принципів ООП, створених інженером Робертом Мартіном (також відомий як Дядько Боб), їх абревіатура звучить як SOLID, де:

— S — принцип єдиного обов’язку (англ. Single responsibility principle), тобто кожна компонента системи (модуль, клас) повинна бути відповідальною тільки за вирішення однієї задачі, а саме, мати лише одну причину для змін. Якщо компонента відповідає за вирішення декількох задач, то інші компоненти, що успадковують її, виявляються тісно пов'язаними один з одним. Тісна зв'язність модулів призводить до складності їх змін при необхідності, так як зміни в одній компоненті тягнуть за собою зміни в іншій;

— O — принцип відкритості/закритості (англ. Open/closed principle), компоненти системи повинні піддаватися розширенню, але не змінам ззовні. Розширення можна проводити через механізм успадкування, що дозволяє створювати нові класи на основі вже існуючих;

— L — принцип підстановки Лісков (англ. Liskov substitution principle), який означає, що класи-нащадки можуть використовуватися замість батьківських класів (часто замість абстракцій, які вони успадковують), не порушуючи роботу програми;

— I — принцип розділення інтерфейсу (англ. Interface segregation principle) — слід створювати спеціалізовані інтерфейси для кожної окремої функції, замість використання одного універсального інтерфейсу. Даний підхід покращує багатошаровість архітектури, дозволяє усунути наслідки, пов'язані з реалізацією великих інтерфейсів, в якому визначено безліч функцій, які не завжди потрібно перевизначати в усіх класах-нащадках. В цьому плані даний принцип дещо нагадує принцип єдиної залежності;

— D — принцип інверсії залежностей (англ. Dependency inversion principle), а саме модулі верхніх рівнів системи не повинні залежати від модулів нижніх рівнів, а якщо точніше, то абстракції не залежить від деталей, а навпаки — деталі повинні залежати від абстракцій.

Дотримання цих принципів дозволяє полегшити тестування системи, так як класи, спроектовані та побудовані за даними принципами, не містять в собі безліч залежностей, а використання абстракцій веде до слабкої зв'язності. Також система залежностей,

спрямована вглиб дозволяє оновлювати реалізацію абстрактного верхнього рівня системи за потреби.

4.2.2 Вибір архітектурного підходу

Для побудови якісної системи на платформі Android рекомендується використовувати різні архітектурні підходи, дотримання яких істотно полегшить створення продукту за допомогою компонент, що надаються самою платформою.

Архітектурні підходи зазнавали множинні зміни протягом останніх кількох років, проте співтовариство розробників Android в 2017 році висунуло на розгляд єдине керівництво з використання архітектурних компонент, які дозволяють створити максимально зручний для підтримання і тестування застосунок, уникаючи написання коду, який повторюється (англ. boilerplate code) в різних модулях системи та не порушуючи принцип розподілу відповідальностей. Спільнота в значній мірі відійшла від монолітної моделі Model-View-Controller (MVC), в якій основними компонентами є модель даних, вигляд (інтерфейс користувача) та модуль управління — контролер, головною функцією якого є отримання та перетворення даних на команди для рівня моделі чи вигляду, на користь більш модульних шаблонів, що піддаються тестуванню, таких як Model-View- Presenter (MVP) і Model-View-ViewModel (MVVM).

Не можна сказати, що певна модель є краща за інші, так як підхід до побудови архітектури цілком і повністю залежить від бізнес-вимог до застосунку, однак для побудови гнучкої архітектури з використанням сучасних компонент фреймворка Android був

обраний архітектурний шаблон Model-View-ViewModel (далі для зручності буде використовуватися абревіатура MVVM).

4.2.3 Бізнес-вимоги

Головною метою даного застосунка є надання допомоги кінцевому користувачеві навчитися управляти своїм часом, таким чином бізнес-вимоги до застосунку були створені на основі різних методик тайм-менеджменту, згаданих в розділах 3.1, 3.2 і 3.3, а саме GTD, метод Помодоро та правило “Don’t break the chain”. Шляхом комбінації методик можна досягти кращої продуктивності з використанням тільки одного застосунку, основними функціями якого є:

— можливість додавати (UC1), редагувати (UC2) та видаляти (UC3) записи;

— змінювати статус завдання (UC4) на виконане;

— переглядати список виконаних завдань (UC5);

— можливість змінювати пріоритет (UC6) або категорію (UC7) завдання вручну після машинного передбачення;

— встановлювати оповіщення (UC8);

— встановлювати таймер на виконання завдання (UC9).

Кожна із наведених бізнес-вимог отримує спеціальне позначення для подальшого використання у процесі аналізу системи, що докладно описаний у розділі 5.

Всі функції можна розглянути на діаграмі прецедентів (англ. Use-case diagram), що знаходиться на рисунку А1 в додатку А, де єдина роль — це користувач системи, а сама система обмежена

прямокутником. Дана діаграма слугує концептуальним представленням системи застосунку і дозволяє інтуїтивно зрозуміти, що користувач може виконувати.

4.2.4 Реалізація шаблону MVVM в застосунку

4.2.4.1 Рівень View

Шаблон MVVM дозволяє відокремити логіку застосунку від візуальної частини. В архітектурі Android уособленнями візуальної частини є такі компоненти як Activity або Fragments, які мають загальну назву Views — це екрани, на яких розташовується графічний інтерфейс, тобто все, з чим надалі буде взаємодіяти користувач. Застосунок може складатися з декількох таких екранів, між якими можна передавати інформацію в обох напрямках в залежності від дій користувача. Точкою входу в застосунок є екран MainActivity, який можна спостерігати на рисунку Б1 в додатку Б, який надалі буде відповідати за відображення списку завдань користувача. Також на рівні View розташовуються такі екрани, як AddEditTodoActivity для додавання або редагування завдання, та TimerActivity для встановлення таймеру.

За розташування елементів в візуальному інтерфейсі відповідають правила, написані в файлі макета для кожного окремого екрану. Для написання даних правил у вигляді текстових полів використовується розширювана мова розмітки — XML (англ. eXtensible Markup Language). Також на цьому екрані розташований елемент контролю — кнопка, при натисканні на яку відбувається перехід до іншого екрану, а саме AddEditTodoActivity. Всі дані для

відображення поставляються на рівень представлення за допомогою спеціальної компоненти — AndroidViewModel, яка знаходиться на рівень нижче, на рівні ViewModel, який детальніше буде описаний у пункті 4.2.4.2.

Компоненти Activity володіють цікавою властивістю: у них є власний життєвий цикл, який вони переживають в залежності від дій користувача, по мірі того, як він переміщається по застосунку. При першому запуску застосунку всі екрани управляються через особливий стек фрагментів, який має назву Back Stack, тобто екран, який є головним на даний момент розміщується наверху цього стеку, а при зміні конфігурації (наприклад, зміна орієнтації екрану мобільного телефону або системної мови) екран видаляється зі стека, що призводить до повного перебудування фрагмента та його відображення з нуля. Цей процес може призвести до великої втрати тимчасових даних користувача, якщо коректно не обробляти схожі ситуації. Саме для збереження даних протягом життєвого циклу фрагмента і всього застосунку потрібно використовувати компонент AndroidViewModel.

4.2.4.2 Рівень ViewModel

Даний рівень є зв’язуючим між рівнями вигляду та моделі та вважається контролером. Будь-який клас, який взаємодіє з графічним інтерфейсом користувача повинен успадковуватися від класу ViewModel на цьому рівні, щоб зберігати і керувати даними на протязі всього життєвого циклу самого фрагмента і застосунка в цілому. В застосунку за це відповідає клас TodoItemViewModel.

Для того, щоб постійно відстежувати дані та відображати їх коректно після внесених змін, необхідно реалізувати шаблон спостерігач (англ. Observer), який дозволяє фіксувати зміни стану об'єкта (в даному застосунку це список справ), який обертається в об'єкт LiveData. LiveData є сховищем даних і дозволяє підписуватися на нього для відстеження змін, причому повідомлення про стан зміни об'єкта буде надсилатися лише активним підписникам, в даному випадку це буде MainActivity, так як саме всередині цього фрагмента відбувається звернення до сховища з даними, які поставляє TodoItemRepository.

4.2.4.3 Рівень Model

В даному шаблоні модель може бути представлена за допомогою сутностей (англ. Entities), які в мові Java є об'єктами, також відомі як "прості старі Java-об'єкти" (англ Plain Old Java Object, POJO), та що зберігаються в базі даних. При роботі з архітектурними компонентами клас сутності анотується спеціальним чином для того, щоб являти собою таблицю бази даних. У застосунку цієї сутністю є об’єкт завдання, тобто окрема одиниця елемента списку. Для звернення до даних також був розроблений окремий шар, який відповідає за звернення до бази даних, так званий репозиторій. У цьому шарі знаходиться клас TodoItemRepository, що є представленням сховища даних, який використовує методи об'єкта доступу до даних (англ. Data access object, DAO) — TodoItemDao. Взагалі репозиторій служить відмінною додатковою абстракцією, так як приховує під собою звернення до різних джерел даних. Наприклад, якщо в ході розробки або розширення програми буде

потрібно перенести дані з бази даних на який-небудь хмарний сервер, то репозиторій буде зберігати в собі реалізацію даного звернення в окремій залежності, яка дозволяє визначати, до якого джерела даних необхідно звернутися для отримання інформації. Рівень моделі тісно пов’язаний із базою даних, тому необхідно ретельно підходити до проектування відповідних абстракцій.

4.2.5 База даних

Основною сутністю для зберігання в базі даних є елемент списку завдань — TodoItem. Так як застосунок не передбачає реєстрації, розумним методом зберігання інформації буде використання бібліотеки Room Persistence library (або просто Room), яка є додатковим рівнем абстракції над прикладним програмним інтерфейсом (англ. Application Programming Interface, API) СУБД SQLite, та яка повністю підтримується платформою Android і є стандартом для зберігання непримітивних значень безпосередньо на смартфоні (примітивними значеннями в даному випадку є будь-які дані в форматі "ключ-значення"). Використання даної бібліотеки дозволяє спростити операції управління даними, які раніше доводилося писати вручну з використанням мови запитів SQL, пропонуючи зручні анотації для базових методів: створення, зчитування, зміна і видалення, також відомими як CRUD-операції (англ. create read update delete), і не тільки. Для виконання нестандартних або складних запитів надається анотація @Query, через яку можна робити запити мовою SQL. Також дана бібліотека дозволяє заносити дані до кеш-пам’яті, коли смартфон не має

доступу до мережі для подальшої синхронізації, якщо вони повинні зберігатися на сервері.

Бібліотека Room передбачає використання трьох головних компонент, які також можна побачити в області Room на рисунку Б1 в додатку Б:

— база даних — клас, який зберігає в собі об'єкт бази даних “todoitems_database” і є точкою взаємодії між рівнями ViewModel і Model. Об'єкт створюється за допомогою шаблону "одинак" (англ. Singleton), що дає гарантії того, що даний об'єкт буде створений всього лише раз під час запуску застосунку, так як залежить від його контексту, і не буде плодити множинні копії при кожному зверненні до бази даних. Також даний клас містить у собі інформацію про сутності, які зберігаються в базі даних, які вказуються заздалегідь за допомогою відповідних анотацій. Тісно взаємодіє з об'єктом доступу до даних TodoItemDao за допомогою використання методів даного інтерфейсу;

— сутність — дана компонента представляє таблицю у базі даних, тобто для кожної сутності має існувати окрема таблиця, в даному додатку це TodoItem, що зберігається в таблиці “todo_items”, із характеристикою та атрибутами якої можна ознайомитися в таблиці 4.1. Типи даних, що відповідають за зберігання дати, часу та логічних значень (boolean) перетворюються в процесі збереження даних за допомогою спеціального класу конвертора Converters, бо за замовчуванням ці типи не мають відповідних значень в СУБД SQLite, тому вони позначені як тип numeric;

— DAO — компонента, що містить методи для доступу до бази даних, представлена інтерфейсом TodoItemDao.

Таблиця 4.1 — Характеристики сутності TodoItem

Сутність	Тип даних	Назва атрибуту	Характеристика
	integer	id	ідентифікатор завдання
	text	title	текст завдання
	text	description	додаткові замітки
	integer	priority	пріоритет завдання
	text	category	категорія
	numeric	has_reminder	індикатор наявності встановленого оповіщення

Продовження таблиці 4.1

	numeric	created_date	дата створення
	numeric	reminder_date	дата оповіщення
	numeric	is_completed	індикатор завершеності завдання

Основний процес звернення до бази даних виглядає як поступове занурення вглиб залежностей, тобто спочатку використовується клас з базою даних TodoItemRoomDatabase, в якому міститься залежність об'єкта DAO — TodoItemDao, за допомогою якого

можна звертатися до самого сховища і працювати з сутністю TodoItem. Графічно процес використання бібліотеки Room зображений на рисунку 4.1.

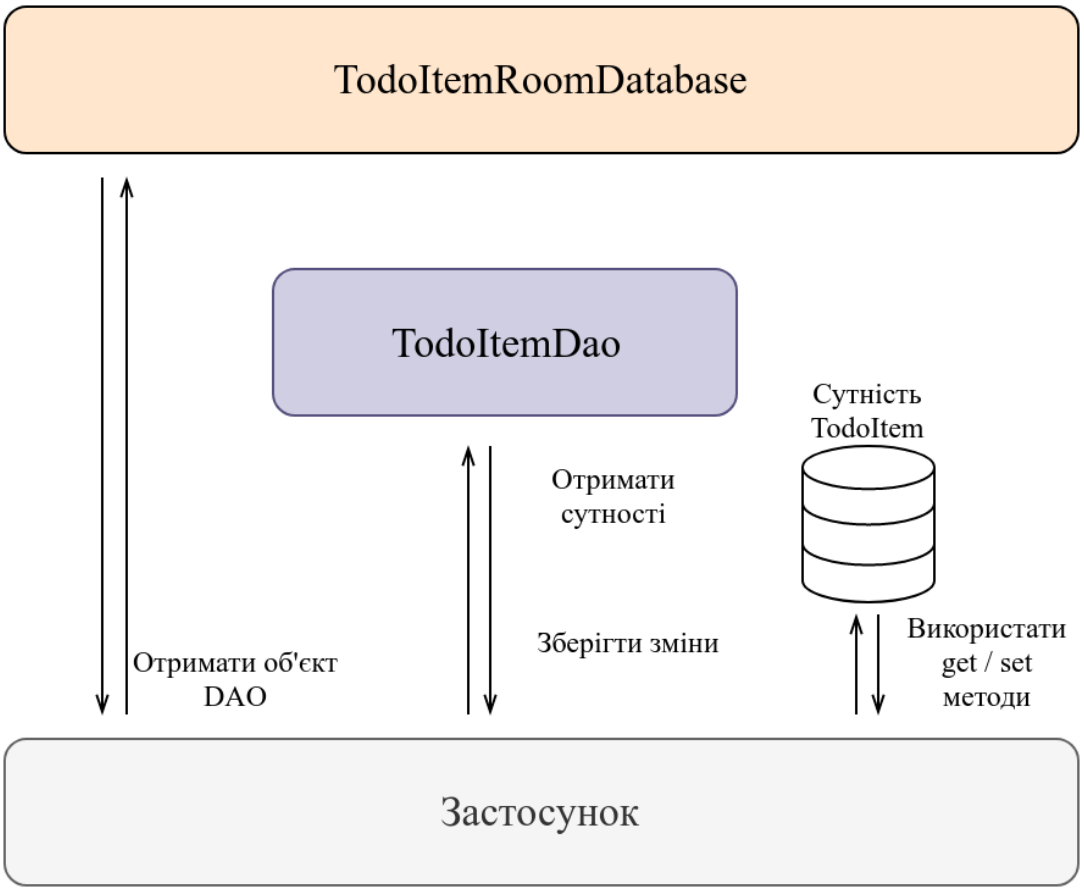

Рисунок 4.1 — Використання бібліотеки Room

4.2.6 Розробка графічного інтерфейсу

Дизайн екранів, з якими взаємодіє користувач в застосунку, ставить собі за ціль зберегти максимальну зручність використання. Для цього кожна компонента була детально побудована як функціонально, так і візуально, грунтуючись на відгуках про схожі існуючі застосунки, розміщені на дистрибутиві Google Play Store. Після аналізу відгуків були виділені такі загальні характеристики застосунків, які отримували найбільш позитивні коментарі:

— мінімалістичний дизайн, що полягає в простоті кольорової гамми і виділенні важливих елементів, які спонукають до дій;

— зручна навігація, схована за замовчуванням;

— неперевантаженість інтерфейсу — відсутність зайвих кнопок й інтуїтивно незрозумілих опцій меню.

Для дотримання принципів мінімалістичного дизайну необхідно дотримуватися головного правила — не використовувати більше трьох кольорів одночасно, тому в якості основного відтінку був обраний темно-синій колір, який викликає довіру, серйозність, може викликати у користувача відданість до справи, якою він займається [17]. Даний відтінок підходить для використання в додатку з управління часом, так як необхідно створити у людини відчуття благополуччя і впевненості в тому, що він робить прогрес. При виборі кольору акценту для деяких елементів був використаний відтінок, що розташовується навпроти синього на колірному колі, а саме помаранчевий, що символізує ентузіазм, професіоналізм, а також є кольором, що відмінно привертає увагу і спонукає користувача до дій.

4.2.7 Машинне навчання та модуль обробки природної мови

На сьогоднішній день штучний інтелект (ШІ) існує в союзі з людським інтелектом в різних областях життя. Можна спостерігати його різні застосування в сферах медицини, науки, торгівлі, інформаційних технологій та інших. У мобільних додатках ШІ використовується найчастіше як особистий помічник, що дозволяє

оптимізувати роботу людину, робити прогнози, засновані на його діях, надавати рекомендації.

Для реалізації помічника під конкретні потреби можуть використовуватися різні алгоритми обробки інформації, що надходить всередину системи такого помічника. Для реалізації інтелектуальної компоненти системи з тайм-менеджменту була обрана така область ШІ, яка називається обробка природної мови (англ. Natural language processing, NLP). Вона належить до області машинного навчання, яка дозволяє реалізувати взаємодію між комп'ютерною і природною людською мовами. Подібні системи використовуються в різних цілях, наприклад для машинного перекладу, виявлення спаму в пошті, або ж розпізнавання мови в режимі реального часу, проте в даному застосунку NLP модуль TextAnalyzer відповідає за обробку тексту завдання, введений користувачем, який на виході привласнює даному тексту одну з чотирьох доступних категорій завдань (work — стосовно роботи, personal — особиста, home — стосовно домашніх справ, health — здоров’я), а також пріоритет виконання, що залежить від дня тижня та отриманої категорії, таким чином допомагає людині в прийнятті рішень за допомогою автоматизації цих дій, в той час як в інших існуючих застосунках подібні дії потрібно виконувати вручну. Із кодом модулю можна ознайомитись в додатку Ґ.

4.2.7.1 Сервіс Google Cloud AutoML Natural Language

Для класифікації контенту, що є текстом завдань, за категоріями використовується сервіс, що надається хмарною платформою від компанії Google, а саме Google Cloud AutoML Natural Language.

Процес аналізу тексту за допомогою алгоритму машинного навчання полягає в подачі на нього великого обсягу текстових даних для обробки, заздалегідь розподілених за категоріями (також відомі як лейбли або класи), а на виході отримається категорія, яка на думку алгоритму найбільш підходить даному тексту. AutoML Natural Language для обробки використовує спосіб "навчання з учителем" (англ. Supervised learning), призначення якого полягає в розпізнаванні певних шаблонів з помічених даних. Даний спосіб навчання дозволяє створити модель, підлаштовану під спеціальні потреби, яка буде розпізнавати тільки ті ділянки текстових даних, які становлять найбільшу значимість, шляхом виявлення закономірностей між текстом і заздалегідь проставленими категоріями.

4.2.7.2 Підготовка даних та тренування моделі

Для онлайн-тренування моделі відповідно до інструкції необхідно докладати не менше 100 прикладів текстових даних на кожну категорію, щоб досягти найбільшої точності в подальших прогнозах. Модель, що використовується в застосунку, проходила тренування на вибірці з 630 примірників з 4 категоріями.

Вибірка містить елементи лише англійською мовою через те, що на даний момент сервіс надає гарантії коректного спрацьовування моделі тільки латиницею. Тренувальні дані були заздалегідь підготовлені і розподілені за категоріями вручну. Після завантаження даних на хмарний сервіс, він автоматично поділяє вибірку на 3 частини:

— 80% від загального обсягу даних відправляються на навчання моделі, так званий тренувальний набір;

— 10% йде на перевірочний набір — це необхідно для перевірки моделі на неупередженість, тобто після процесу навчання на тренувальних даних модель в кінцевому підсумку може висувати передбачення, ґрунтуючись на тренувальному наборі, однак щоб отримати неупереджене прийняття рішень на більш складних і неоднозначних текстових прикладах необхідно використовувати перевірочний набір;

— 10% залишкових даних відправляються на аналіз всередину алгоритму, який до цього пройшов етап навчання і пристосування на перевірочному наборі.

4.2.7.3 Процес класифікації даних

Класифікація в навчанні з учителем відбувається за допомогою методу логістичної регресії, що дозволяє моделі передбачити ймовірність приналежності вихідних даних певному класу. У додатку існує 4 класи, що знаходяться у довільному порядку, тому в даному випадку буде використовуватися поліноміальна логістична регресія (бо кількість класів > 2), яка на виході обчислень прогнозує ймовірність приналежності змінної до класу у вигляді значення в інтервалі [0, 1], на відміну від лінійної регресії, де результат — безперервний набір чисельних значень, який може розміщуватися на інтервалі [-∞, +∞]. Однак, щоб трактувати це значення ймовірності в плані приналежності або неналежності змінних одному з класів необхідно ввести поняття порога прийняття рішення (англ. Classification threshold, Decision threshold)

— це деяка межа, орієнтуючись на яку можна передбачити приналежність тексту певній категорії. Часто при навчанні моделі це значення приймається як 0.5 і потім налаштовується в залежності від отриманих результатів. Приклад, в якому модель вірно передбачила категорію work для текстових елементів при порозі 0.5 можна побачити на рисунку 4.2. Числові значення поруч із лініями це значення ймовірності приналежності елементу до класу.

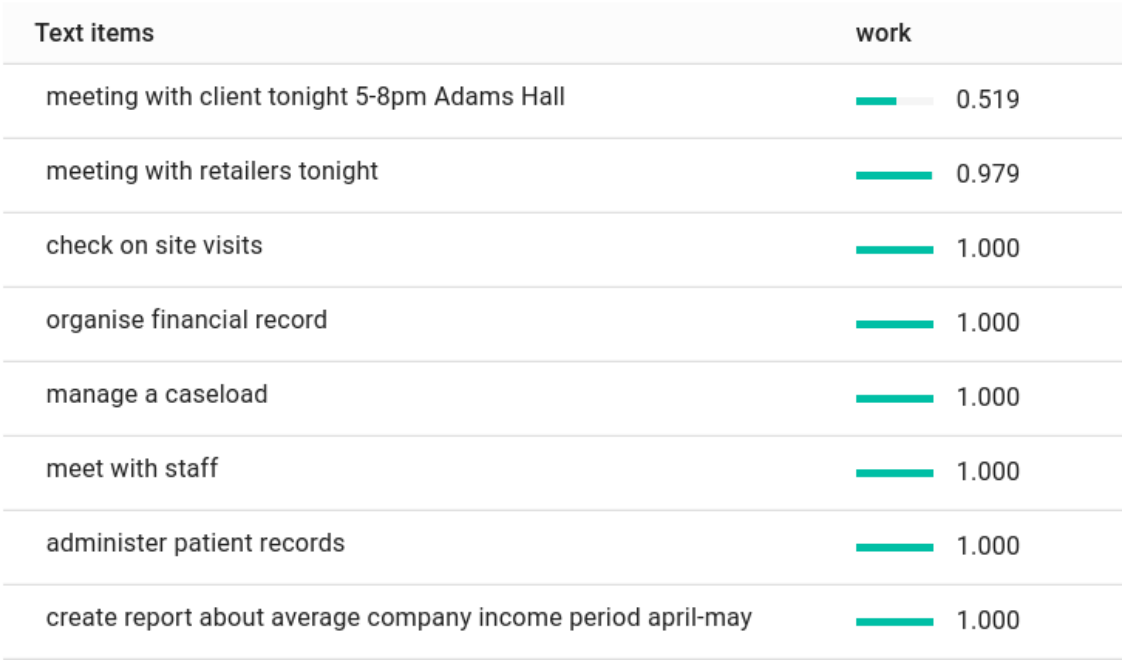

Рисунок 4.2 — Передбачення моделі після тренування

4.2.7.4 Визначення та виставлення пріоритету завдання

Процес виставлення пріоритету завданням виконується спеціальним методом після попередньої обробки тексту для отримання категорії, за який відповідає NLP модуль, описаний в розділі 4.2.7. На вхід в метод подається елемент зі списку завдань, що представляється об’єктом сутності todoItem. Для цілісності картини процес вибору категорії зображений на блок-схемі в додатку В. Для простоти зображення буде вважатись, що категорія

— це category, день тижня — createDate отримані за допомогою get-методів об’єкта, а пріоритет — priority, виставлений через set-метод.

Визначення пріоритету завдання відбувається в першу чергу у залежності від дня його створення. У кожного дня тижня є порядковий номер в інтервалі [1, 7]. Якщо порядковий номер знаходиться в інтервалі [1, 5], це означає будній день та людині слід приділити увагу справам, які стосуються його робочих обов'язків або здоров'я (наприклад відвідування тренажерної зали), так як в ці дні особистого часу після роботи залишається досить мало. Порядкові номери 6 та 7 відповідають за вихідні дні, відповідно для збереження психічного балансу та уникнення перевтоми слід сконцентруватися на особистих і домашніх справах, а не присвячувати себе роботі. В даному випадку категорія "здоров'я" буде мати пріоритет 3, однак це не означає, що людина не може вручну виставити пріоритет, як їй заманеться.

Висновки: в ході проектування і реалізації застосунку були використані кращі практики проектування, які дозволяють будувати надійну систему, яку зручно тестувати, з використанням відповідних компонент, які надаються фреймворком платформи Android. Бізнес-вимоги були сконструйовані з метою вирішення існуючої проблеми управління часом, в результаті застосунок володіє базовим необхідним функціоналом, до того ж для автоматизації деяких функцій був застосований алгоритм обробки природної мови, який допомагає користувачеві в розстановці пріоритетів на основі заздалегідь передбачених категорій завдань.

5 АНАЛІЗ ГОТОВОГО ПРОГРАМНОГО ПРОДУКТУ

5.1 Модульне та інтеграційне тестування

Тестування застосунка є важливим етапом розробки — таким чином система перевіряється на правильність спрацьовування методів в конкретних сценаріях і відповідність бізнес-вимогам. Модульне тестування, також відоме як юніт-тестування, є одним з найпопулярніших видів відлагодження додатків, так як даний підхід дозволяє перевірити на коректність найчастіші за використанням модулі в системі. В контексті тестування модулем прийнято вважати найменшу частину програми, яку можна протестувати — тобто окремий метод. В архітектурі Android графічний інтерфейс користувача грає важливу роль, тому щоб протестувати його на коректність взаємодії із користувачем необхідно також виконувати автоматизоване інтеграційне тестування за допомогою певних фреймворків, що дозволяють відтворити дії, які може виконувати людина, використовуючи застосунок. Найбільш розповсюдженими у використанні фреймворками для тестування мобільних застосунків є jUnit, hamcrest та Espresso.

Для аналізу покриття додатку тестами, точності їх спрацьовування і їх доцільності необхідно розробити тестові сценарії, кожен із яких буде відповідати вимогам, визначеним у розділі 4.2.3, які потім будуть використовуватися в матриці відповідності цим вимогам.

Каркас тестових сценаріїв наведено в таблиці 5.1. Лістинг класу з тестами до відповідних тест-кейсів знаходиться в додатку Д.

Таблиця 5.1 — Таблиця тестових сценаріїв

Тест-кейс	Назва	Очікуваний результат
TC1	Перевірка методу додавання запису	Запис відображається на екрані зі списком завдань
TC2	Перевірка методу редагування запису	Запис змінений та відображається на екрані зі списком завдань
TC3	Перевірка методу видалення запису	Запис видалено та не відображається на екрані зі списком завдань
TC4	Перевірка методу зі зміни статусу завдання	Запис відмічений як виконаний та відображається на екрані зі списком завдань
TC5	Перевірка методу відображення списку завдань	Список завдань відображується на екрані
TC6	Перевірка методу зміни пріоритету	Пріоритет запису змінено, запис відображається на екрані списку завдань
TC7	Перевірка методу зміни категорії	Категорію запису змінено, запис відображається на екрані списку завдань
TC8	Перевірка методу	Текст із встановленим оповіщенням відображається на екрані додавання або редагування завдання

	налаштування оповіщень	

Продовження таблиці 5.1

ТС9	Перевірка методу встановлення таймера	Екран із таймером завантажений

Матриця відповідності вимогам (англ. Requirements Traceability Matrix, RTM) являє собою двовимірну таблицю і дозволяє упевнитися в тому, що кінцевий продукт відповідає вимогам, поставлених потенційним кінцевим користувачем та розробником, і не має будь-яких недоліків або слабких місць. У заголовках колонок таблиці розташовані вимоги, а в заголовках рядків — тестові сценарії. На перетині — відмітка, що означає, що вимога поточної колонки покрита тестовим сценарієм поточного рядка. Матриця зображена у вигляді таблиці в додатку Г і демонструє покриття функціоналу застосунку тестами.

5.2 Метрики оцінки продуктивності моделі

Для оцінки моделі необхідно розглядати такі характеристики, як загальна точність, що складається із чіткості і повноти. Сервіс AutoML Natural Language надає можливість отримання даних метрик як у текстовому, так і в графічному вигляді, надаючи графік ROC-кривої (англ. Receiver operating characteristic, робоча характеристика приймача), що зображує ефективність класифікації

моделі при різних значеннях порога прийняття рішень, описаного в розділі 4.2.7.3.

5.2.1 Матриця помилок

Будування матриці помилок (англ. Confusion matrix, Error matrix) дозволяє зобразити ефективність алгоритма у вигляді матриці 2x2. На головній діагоналі зазвичай розміщують позитивний результат спрацьовування моделі, а саме:

— True Positive (TP) — модель правильно передбачила очікуваний клас;

— True Negative (TN) — модель правильно передбачила неочікуваний клас.

На зворотній діагоналі розташовуються негативні результати:

— False Positive (FP) — модель неправильно передбачила очікуваний клас;

— False Negative (FN) — модель неправильно передбачила неочікуваний клас.

5.2.2 Точність моделі

Для отримання кількісного результату точності моделі використовується матриця помилок, описана у попередньому розділі. Для кращого розуміння у таблиці 5.2 зображена матриця помилок для моделі на прикладі передбачень відношення тексту завдання “організувати фінансовий звіт” до класу ‘work’. За очікуваний клас береться ‘work’, неочікуваний — ‘health’.

Таблиця 5.2 — Матриця помилок для класу ‘work’

True Positive (TP): Очікуваний клас: ‘work’ Модель передбачила: ‘work’	False Positive (FP): Очікуваний клас: ‘health’ Модель передбачила: ‘work’
False Negative (FN): Очікуваний клас: ‘work’ Модель передбачила: ‘health’	True Negative (TN): Очікуваний клас: ‘health’ Модель передбачила: ‘health’

Результати, отримані в TP та TN відносяться до правильних передбачень, тому що обидва передбачених класи є очікуваними, а FP та FN до неправильних. Таким чином загальна точність моделі (англ. Average precision) розраховується згідно наступної формули:

$$\text{Точність} = \frac{\text{кількість правильних передбачень}}{\text{загальна кількість передбачень}} = \frac{TP + TN}{TP + TN + FP + FN} \qquad (5.2.2.1)$$

Дана формула знадобиться у подальших розрахунках для показу залежності чіткості (англ. Precision) від повноти (англ. Recall), але для початку треба вивести поняття цієї метрики.

5.2.3 Повнота та чіткість моделі

Значення чіткості не завжди показує ефективність передбачення моделі, тому що вона показує відношення кількості правильних

передбачень до загальної кількості позитивних передбачень [18] і розраховується за наступною формулою:

$$\text{Чіткість} = \frac{TP}{TP + FP} \qquad (5.2.3.1)$$

Такий підхід залишає позаду повноту картини: відкидання помилкових спрацьовувань (TN та FN) дає помилку у передбаченнях, тому вводиться поняття повноти, яка відповідає на питання “яка доля передбачень була розрахована правильно?”. Таким чином повнота показує відсоток класів, що були ідентифіковані правильно серед усіх передбачень, а не лише серед позитивних. Формула повноти буде виглядати наступним чином:

$$\text{Повнота} = \frac{TP}{TP + FN} \qquad (5.2.3.2)$$

Загальна точність моделі при порозі прийняття рішень 0.75 для кожного з 4 класів наведена у таблиці 5.3. Значення порога рішень було обрано з метою показати точність моделі при реальному передбаченні, нижче значення даного порога дозволяє отримати вищу повноту передбачень, але знижує точність.

Таблиця 5.3 — Загальна точність моделі

	Клас ‘work’	Клас ‘health’	Клас ‘home’	Клас ‘personal’
Чіткість, %	100	97.4	100	88.9
Повнота, %	96	97.4	100	100

Загальні значення середньої точності, чіткості та повноти моделі зображені на рисунку 5.1.

Рисунок 5.1 — Показники ефективності моделі

Залежність чіткості та повноти моделі від порога прийняття рішень зображені на рисунку 5.2.

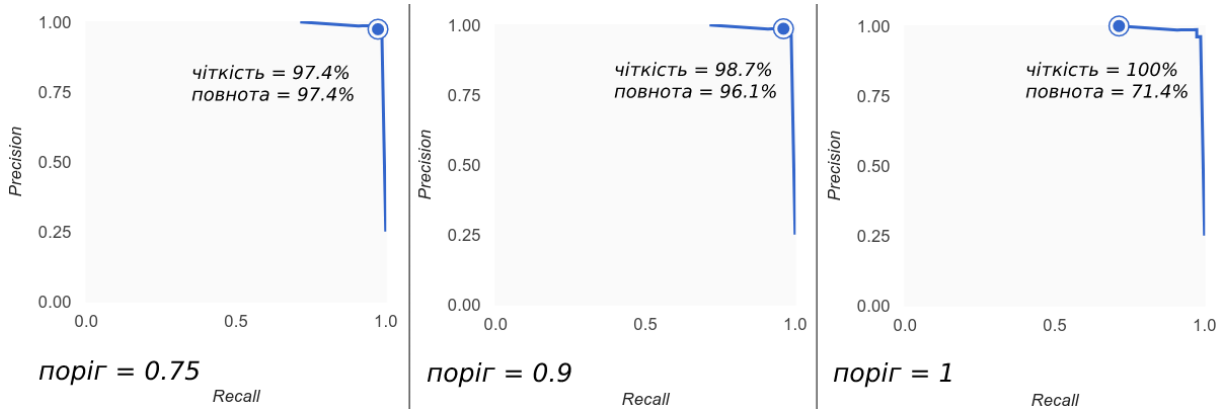

Рисунок 5.2 — Залежності метрик від порога прийняття рішень

Висновки: після проведення аналізу готового застосунку, а саме використаної моделі та його покриття тестами, можна стверджувати, що реалізована система відповідає поставленим бізнес-вимогам, а тестування покриває відповідний функціонал. Інтелектуальна компонента показує досить високу ефективність, а метрики хмарного сервісу дозволяють відстежувати найменші неточності у роботі моделі і при необхідності підлаштовувати її під нові потреби.

ВИСНОВКИ

Головною метою при виконанні дипломного проекту було дослідження залежності продуктивності людей і їх фізичного та ментального благополуччя від усвідомлення цінності часу та вмінні ним керувати. Аналіз проблеми показав, що від невміння або ж відсутності знань про існуючі методики планування часу люди схильні потрапляти в стресові стани, що може послужити початком різних розладів і опустити загальну ефективність на досить низький рівень аж до того, що людина усвідомлено відкладає виконання завдань на самий крайній термін здачі, що призводить до перевтоми організму. Саме для уникнення подібних результатів існують методики з тайм-менеджменту. Практичне застосування цих методик має місце в світі мобільних застосунків, які на сьогоднішній день є найкращими кишеньковими помічниками людини при грамотному використанні і дозволяють структурувати інформацію і направляти розумові процеси в інше русло.

Після дослідження проблеми було прийнято рішення в створенні мобільного застосунка, який відповідав би вимогам сучасного користувача, мав необхідний функціонал і володів лаконічним дизайном. Платформою для реалізації задуманого стала мобільна платформа Android. Головним виконавчим модулем, який допомагає людині приймати рішення і виставляти пріоритети став алгоритм обробки природної мови, що аналізує текст завдань, введений людиною, і на основі цього висуває оптимальну пропозицію з приводу переваги на користь виконання того чи іншого завдання.

Головною гіпотезою було поліпшення загального самопочуття співробітника після тимчасового користування подібним застосунком, проте через відсутність реальних користувачів проведення практичних

експериментів на даному етапі не є можливим. В цілому, ефективність алгоритму була підтверджена проведенням аналізу моделі з використанням статистики на хмарному сервісі, проте варто зауважити, що це не є емпіричним дослідженням, тому для повноти картини необхідні експерименти шляхом впровадження готового продукту на підприємства з метою отримання конкретних результатів, які показали б точність спрацьовування моделі в різних ситуаціях.

Готовий застосунок в майбутньому можна поліпшити шляхом додавання нового функціоналу, наприклад можливість створювати списки на тиждень вперед, а не тільки на поточний день, проводити інтеграцію з різними сервісами, такими як Google календар або ж імпортувати події з Facebook з використанням відповідних прикладних програмних інтерфейсів, можливість змінювати мову інтерфейсу та обробляти текст на різних мовах. Також можна додати можливість пропонування завдань з різними категоріями на основі вже створених або ввести ввести опцію створення відгуку у кінці дня, щоб аналізувати стан користувача і його рівень задоволення поточним станом справ для створення рекомендацій щодо активностей.

На останок варто зазначити, що проблема тайм-менеджменту була, є, і буде актуальною довгий час в зв'язку з ростом розвитку технологій, тому кількість обов'язків людини завжди буде рости для того, щоб відповідати новим вимогам. З одного боку це може бути вирішено шляхом впровадження на підприємства і фірми різних інтелектуальних помічників для автоматизації завдань людини, проте в тих областях, де необхідна її безпосередня участь або ж де людський фактор відіграє дуже важливу роль, потрібно сконцентрувати сили на допомозі людині в рішенні такого складного завдання, як управління власним часом і усвідомлення його цінності.

ПЕРЕЛІК ПОСИЛАНЬ

1. Lucius Annaeus Seneca Ad Paulinum, De brevitate vitae. Wikipedia [Електронний ресурс] : Режим доступу: https://en.wikisource.org/wiki/On_the_shortness_of_life — On the shortness of life. 13.10.2018 p.

2. Stephen Chamberlain Chronos and Kairos: The Gods of Time [Електронний ресурс] : Режим доступу: https://www.stephenchamberlain.net/chronos-and-kairos-the-gods-of-time — Chronos and Kairos: The Gods of Time. 07.01.2019 p.

3. Peter Meerlo, Arianna Novati [Електронний ресурс] : Режим доступу : https://www.rug.nl/news/lack_of_sleep_causes_hippocampus_to_shrink — Lack of sleep causes hippocampus to shrink. 15.09.2017 p.

4. Bill Hathaway [Електронний ресурс] : Режим доступу : https://news.yale.edu/yale-study — Even in the healthy, stress causes brain to shrink, Yale study shows. 09.01.2012 p.

5. Marianna Virtanen BMJ 2015;350:g7772 [Електронний ресурс] : Режим доступу : https://www.bmj.com/content/350/bmj.g7772 — Long working hours and alcohol use: systematic review and meta-analysis. 13.01.2015 p.

6. Maia Szalavitz [Електронний ресурс] : Режим доступу : http://healthland.time.com/2012/03/05/decision-making-under-stress — Decision-Making Under Stress: The Brain Remembers Rewards, Forgets Punishments. 15.03.2012 p.

7. Sarah Stodola [Електронний ресурс] : Режим доступу : http://mentalfloss.com/article/63887/procrastination-through-ages — Procrastination Through the Ages: A Brief History of Wasting Time. 11.05.2015 p.

8. Piers Steel The Nature of Procrastination: A Meta-Analytic and Theoretical Review of Quintessential Self-Regulatory Failure [Електронний ресурс] : Режим доступу : http://studiemetro.auinstallation29.cs.au.dk/fileadmin/www.studiemetro.au.dk/Procrastination_2.pdf — University of Calgary, Psychological Bulletin Vol.133, No. 1, 2007.— 30 с.

9. Lily Herman Joseph Ferrari Infographic [Електронний ресурс] : Режим доступу : https://www.themuse.com/advice/how-to-defeat-procrastination — How to Defeat Procrastination Based On Your Personality Type. 22.05.2019 р.

10. Benjamin Franklin To Joseph Priestley. Wikipedia [Електронний ресурс] : Режим доступу : https://en.wikipedia.org/wiki/Benjamin_Franklin#Decision-making — Benjamin Franklin. 30.05.2019 р.

11. Метод Pomodoro. Wikipedia [Електронний ресурс] : Режим доступу : https://uk.wikipedia.org/wiki/Pomodoro — Метод Pomodoro. 21.04.2019 р.

12. Nick Babich Best Practices For Mobile Design [Електронний ресурс] : Режим доступу : https://www.smashingmagazine.com/2018/08/best-practices-for-mobile-form-design/ — Best Practices For Mobile Design. 28.08.2018 р.

13. Statista [Електронний ресурс] : Режим доступу : https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ — Number of smartphone users worldwide from 2014 to 2020 (in billions). 27.05.2019 р.

14. Newzoo [Електронний ресурс] : Режим доступу : https://newzoo.com/insights/rankings/top-50-countries-by-smartphone-penetration-and-users/ — Top 50 Countries/Markets by Smartphone Users and Penetration. 27.05.2019 р.

15. Sarah Perez Report [Електронний ресурс] : Режим доступу : https://techcrunch.com/2017/05/04/report-smartphone-owners-are-using-9-apps-

per-day-30-per-month/ — Report : Smartphone owners are using 9 apps per day, 30 per month. 27.05.2019 p.

16. Digital Trends [Електронний ресурс] : Режим доступу : https://www.digitaltrends.com/mobile/android-vs-ios/ — Android vs iOS: Which smartphone platform is the best? 17.05.2019 p.

17. Aidan Huang Anatomy of Colors in Web Design [Електронний ресурс] : Режим доступу : https://onextrapixel.com/anatomy-of-colors-in-web-design-blue-and-the-cool-look/ — Anatomy of Colors in Web Design: Blue and the Cool Look. 22.01.2010 p.

18. Classification: Precision and Recall. Machine Learning Google Developers [Електронний ресурс] : Режим доступу : https://developers.google.com/machine-learning/precision-and-recall — Classification: Precision and Recall. 05.03.2019 p.

ДОДАТОК А

Smart Manager

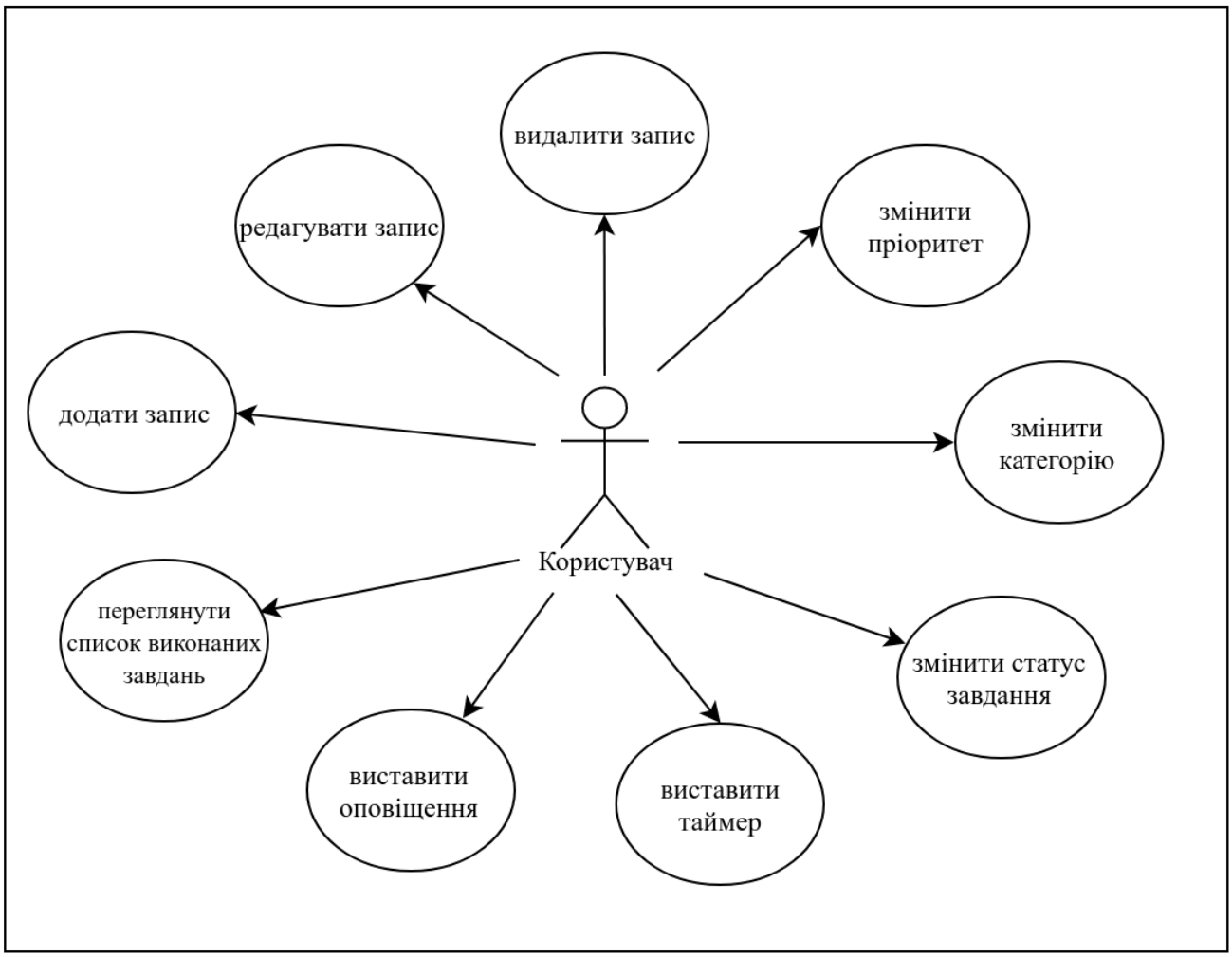

Рисунок А1 — Діаграма прецедентів системи

ДОДАТОК Б

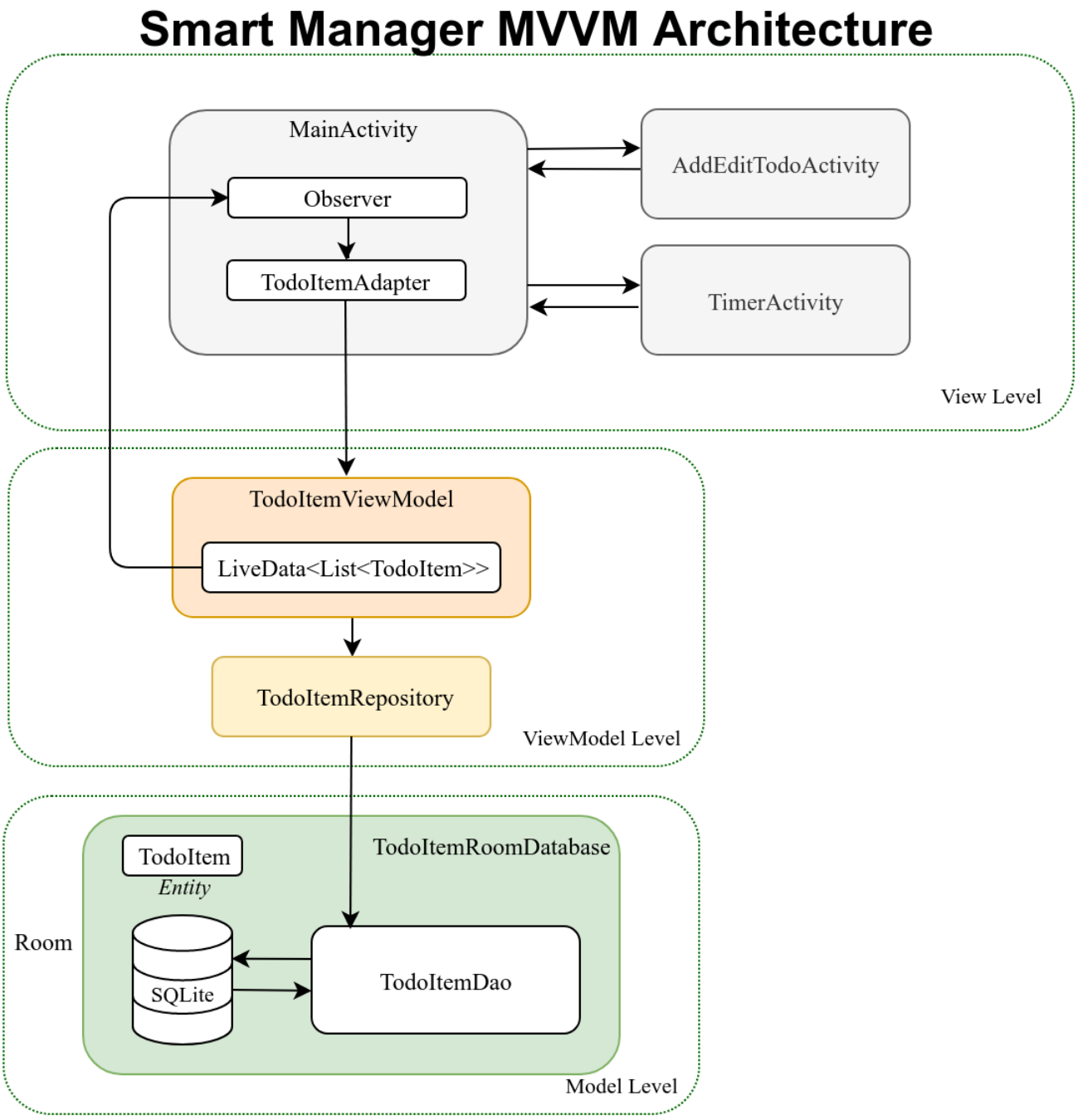

Рисунок Б1 — Реалізація шаблону MVVM в застосунку

ДОДАТОК В

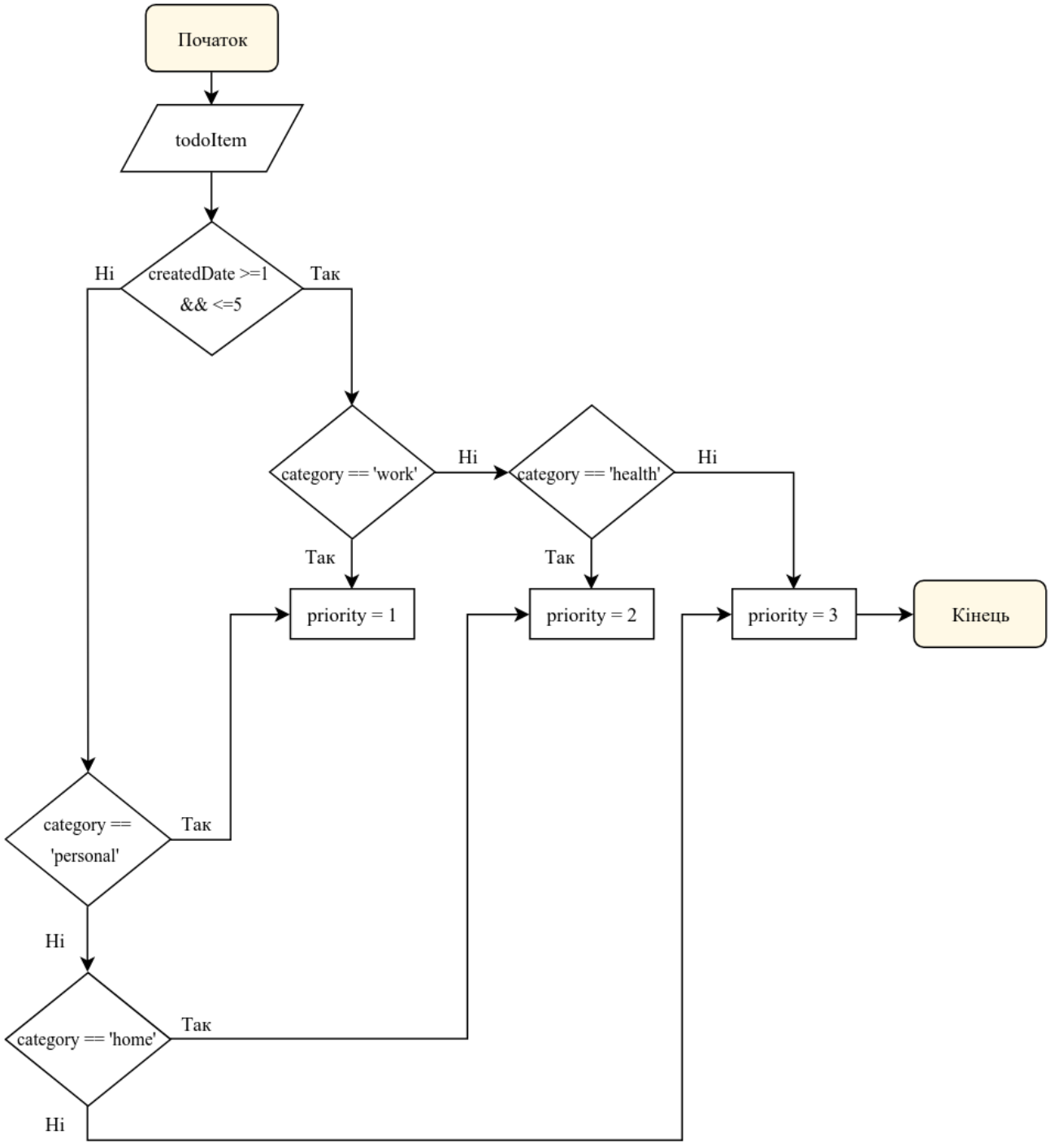

Рисунок В1 — Блок-схема визначення пріоритету завдання

ДОДАТОК Г

Таблиця Г1 — Матриця відповідності вимогам

UC / TC	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9
TC1	X								
TC2		X							
TC3			X						
TC4				X					
TC5					X				
TC6						X			
TC7							X		
TC8								X	
TC9									X

ДОДАТОК Ґ

ДОДАТОК Д